04

Como conectar una aplicación a una base de datos

Transcripción

[00:00] Hola, ya configuramos nuestra base de datos y realizamos algunas operaciones en ella desde una terminal. Ahora, vamos a empezar a desarrollar nuestra aplicación para que se conecte a esta base de datos y realice las mismas operaciones de insertar, modificar, borrar y buscar registros.

[00:17] Pero antes de ir al código, tenemos que entender cómo funciona la conexión de aplicaciones Java a servicios de base de datos. En este caso, nosotros tenemos aquí en este cuadro una base de datos de MySQL. Ya ella está ejecutando ahí en un servidor, con un protocolo único de comunicación.

[00:39] ¿Y nosotros cómo vamos a poder hacer que nuestra aplicación aquí Java pueda conectarse a esta base de datos? Bueno, a ver. Que sería esta pelotita de acá, que va a facilitar la conexión para nosotros. Para estos tipos de operaciones podemos recurrir a un recurso muy común en el desarrollo de aplicaciones, que es la utilización de librerías.

[01:16] El equipo de MySQL ha creado una librería Java que facilita la vida de quien está creando una aplicación y realiza toda la comunicación entre la aplicación y la base de datos. La lib va proveernos una interfaz de comunicación que nuestra aplicación Java a lograr conectarse con el protocolo de comunicación de la base de MySQL.

[01:36] Entonces aquí tendríamos, en lugar de esta pelotita, una herramienta que sería esta librería de MySQL. Y aquí voy a poner un nombre, librería

MySQL. Con eso nos conectamos ahí a la base de datos entre la aplicación y la base de datos. Tenemos esa librería en el medio.

[02:00] En Java nosotros cuando queremos importar una librería, nosotros tenemos que importar un archivo .jar de la misma. Es todo el proyecto de la librería compilado en un archivo .jar.

[02:13] Nosotros podemos hacerlo manualmente, vamos ahí en el Eclipse, importar archivo, vamos a una carpeta y lo importamos a nuestro proyecto, pero también podemos estar utilizando una herramienta de gestión de dependencias como el Maven, el Gradle y cualquier otro que puedan conocer.

[02:30] Estas librerías .jar contienen un conjunto de operaciones comunes que pueden ser utilizadas en cualquier aplicación para una finalidad específica. En este caso, una librería de MySQL para hacer la conexión entre la aplicación y la base de datos, pero podríamos tener una librería de operaciones matemáticas también, es otro ejemplo.

[02:50] En nuestro caso, como había dicho, la librería va a estar haciendo la conexión entre la aplicación y la base de datos. Esas librerías que hacen este tipo de operación, la librería de MySQL, ellas son conocidas como un driver, entonces lo que estamos haciendo aquí en nuestro proyecto es agregar un driver de conexión de MySQL que nos va a ayudar con este tipo de operación de conexión a la base.

[03:17] Entonces, en lugar de librería MySQL, voy a decir que es un driver de MySQL. Ahora sí, el nombre está correcto. Así está. Ahora imaginemos que nuestra aplicación va creciendo y nosotros empezamos a integrarnos con más de una base de datos y aquí tenemos la de MySQL que está aquí. Voy a cambiar un poco la flecha.

[03:44] Y voy a duplicar esta parte para decir que ahora vamos a estar conectando también, no solamente con ella, pero con una segunda base de datos, entonces digamos que vamos a estar aquí ahora utilizando el SQL Server.

Y tenemos el driver de SQL Server que también está agregado a nuestra a nuestra aplicación.

[04:10] Bueno, entonces tenemos dos drivers de conexión de base de datos y si van creciendo más bases de datos aquí, van entrando a la aplicación, tenemos que ir agregando más drivers. Pero hay un otro lado también, que nosotros podemos cambiar la base de datos. Es decir, mira ahora en lugar de estar utilizando MySQL y SQL Server, nosotros vamos a cambiar eso para utilizar el Oracle.

[04:32] ¿Y cómo queda eso? ¿Qué consecuencias tenemos con esta situación? Una de las consecuencias es que vamos a tener que cambiar el driver de conexión de la base de datos. Todo lo que tuvimos desarrollado para conectarse con MySQL y SQL Server, ahora vamos a tener a tener que salir cambiando para conectarse solamente con Oracle.

[04:55] Y eso es un escenario que va a generar un montón de trabajo de refactorización. Porque para MySQL nosotros tenemos una interfaz específica que nos ayuda a conectarnos con la base de datos y esta interfaz contiene sus métodos específicos también, como por ejemplo aquí, en el driver MySQL nosotros tenemos la interfaz MySqlConnector que tiene el método getConnection.

[05:28] Y de esta forma nosotros tomamos una conexión a la base de datos desde nuestra aplicación, pero aquí en el SQL Server, nosotros tenemos una otra forma de llamar y de buscar una conexión que sería con SQL ServerConnectionProvider.connect. Ya son dos métodos distintos de dos interfaces distintas.

[05:55] ¿Entonces vieron la confusión que queda? Vamos a tener que parar un poquito para entender mejor este problema, vamos a olvidar un poco la base de Oracle ahora y vamos a pensar acá en esas dos bases de datos. Digamos que vamos a tener que elegir una de las dos, y en lugar de MySQL vamos a estar

trabajando ahora con SQL Server y todos los demás SQL nosotros vamos a tener que salir migrando.

[06:20] Con eso nosotros tenemos que salir cambiando todas las partes que tenemos que tomar la conexión con la base de datos de MySQL para estar utilizando ahora la de SQL Server. Parece sencillo, pero si volvemos a cambiar la base una vez más y pensamos en poner una tercera base y salir cambiándolo todo otra vez, tenemos que salir cambiando todo eso una vez más.

[06:46] Y siempre cada vez que venga una otra base de datos o una otra necesidad, tenemos todo este trabajo para hacer. ¿Y cómo solucionamos esta situación? Por suerte, este lenguaje maravilloso llamado Java tiene una capa de abstracción que nos facilita la vida para conectarnos a cualquier driver de base de datos que vengamos a utilizar. Esta capa va a quedar aquí.

[07:10] Voy a borrar estas flechitas voy a poner una herramienta más acá para acomodar un poquito más. Y esta capa va a quedar aquí entre la aplicación Java y los drivers de MySQL y también de SQL Server. Entonces, esta capa va a estar utilizando, va a estar realizando la comunicación entre la aplicación y los distintos drivers que lleguemos a tener aquí.

[07:42] Si llega un tercer driver, es esta abstracción que va a estar haciendo la comunicación para nosotros. Esta capa tiene un nombre. Se llama JDBC de Java Database Connectivity. El JDBC es una lib que contiene las interfaces de comunicación y operación con la base de datos.

[08:07] Y los drivers específicos para cada base de datos implementan a estas interfaces. Entonces, el driver de MySQL y el driver de SQL Server y el driver de cualquier otra base de datos que tenga una librería para Java, todas sus clases implementan las interfaces de JDBC.

[08:27] Con eso, solo con la abstracción de JDBC vamos a poder conectarnos a cualquier base de datos sin tener que cambiar ninguna línea de código.

Entonces venga lo que sea de base de datos acá, con la utilización de las interfaces, nosotros no tenemos que salir refactorizando nada.

[08:46] Y ahora aquí, con el JDBC, si queremos conectarnos a una base de datos vamos a estar utilizando la clase DriverManager y el método getConnection. Así, aquí en este método vamos a estar enviando los parámetros que son la URL de conexión con la base de datos, el usuario y la contraseña.

[09:09] En la URL nosotros tenemos un formato distinto para salir, para hacer la configuración. El formato de la URL es el siguiente. Voy a agrandar un poquito más acá del tamaño de esta fuente así la ven bien. Aquí está. El formato es el siguiente: jdbc, dos puntos, el tipo de la base de datos por ejemplo, MySQL, dos puntos, barra, barra. Ahí decimos la URL de la conexión que sería en este caso local host:3306 si no me equivoco, es el puerto de MySQL, barra, el nombre del esquema.

[09:53] Entonces tenemos aquí, sería por ejemplo la base de pruebas: base_de_pruebas. Y podemos agregar aquí también parámetros opcionales. En este caso, no tenemos ningún parámetro. Este sería un ejemplo de la URL de la base de datos. Ahora que sabemos cómo tomar la conexión con la base de datos, podemos realizar las operaciones que necesitemos en ella.

[01:21] Y por fin este es el dibujo de cuáles son las capas de comunicación que hacen que nuestra aplicación llegue a la base de datos de forma sencilla. Parece complejo, pero es solo la primera impresión. En la próxima clase vamos a crear un proyecto para poner en práctica lo que recién aprendimos y ver cómo es súper sencillo.